9. Lucy

Program Name: Lucy.java Input File: lucy.dat

Lucy is an artist. Her preferred medium is ASCII characters. She creates large images using nothing more than the characters she finds on her computer's keyboard. She stores these images in data files and sends them to a company called ***Print This!*** to be printed on canvas. Lucy then frames the prints and puts them on sale in her booth down on Guadalupe street in Austin. A problem has arisen with sending the files to ***Print This!*** Lucy's artwork has become so big that it is taking too much time and using too much bandwidth to send the files to the printer. Because you are one of Lucy's most treasured friends and she knows that you are a skilled programmer she has asked if you can write a program that will compress her files and make them easier to send to the printer.

A quick check of the internet turns up an algorithm that looks like it will do the job. It is called **run-length encoding**. Run-length encoding is a very simple "loss-less" data compression routine in which runs of characters (characters that are repeated in sequence) are stored as a pair of values, the character itself and the number of times it occurs in succession. This algorithm works best on data that contains many such runs, simple graphic images such as icons, line drawings, and animations. This algorithm is just what we need to compress Lucy's art work without any loss of each image. Run-length encoding for the string `aaaiiOOOOOOqwwwww**********` would result in `a3i2O6q1w5*10`.

Input: The first line of data will contain a value D representing the number of drawings to compress, followed by D drawing data set. Each drawing data set consists of a number N on one line representing the number of lines in the drawing, followed by N lines that make up the drawing. Each line will contain non-digit alpha and symbol characters, including spaces..

Output: For each of the drawings, N lines that consist of the run-length encoding for that line in the original drawing. Each encoded drawing should be ended by this line, "=====".

Sample input:
```
2
13
....................
@..................@
.@................@.
..@..............@..
...@@@@@@@@@@@@@@...
...@............@...
....@..**..**..@....
....@..**..**..@....
.....@........@.....
..../@...++...@.....
.../&&@..++..@......
../&&&/@@@@@@.......
./&&&/..............
8
..............................
....____. ....................
....|    |____ ___  _______...
....|    \__  \\  \/ /\__  \..
/\__|    |/ __ \\   /  / __ \_
\________(____  /\_/  (____  /
..............\/...........\/.
..............................
```

(Lucy – cont)

Sample output:
```
.20
@1.18@1
.1@1.16@1.1
.2@1.14@1.2
.3@14.3
.3@1.12@1.3
.4@1.2*2.2*2.2@1.4
.4@1.2*2.2*2.2@1.4
.5@1.8@1.5
.4/1@1.3+2.3@1.5
.3/1&2@1.2+2.2@1.6
.2/1&3/1@6.7
.1/1&3/1.14
=====
.30
.4_4.1 1.20
.4|1 4|1_4 1_3 2_7.3
.4|1 4\1_2 2\2 2\1/1 1/1\1_2 2\1.2
/1\1_2|1 4|1/1 1_2 1\2 3/1 2/1 1_2 1\1_1
\1_8(1_4 2/1\1_1/1 2(1_4 2/1
.14\1/1.11\1/1.1
.30
=====
```